(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **09069060 A**

(43) Date of publication of application: **11 . 03 . 97**

(51) Int. Cl **G06F 12/00**

(21) Application number: **07223824**

(22) Date of filing: **31 . 08 . 95**

(71) Applicant: **TOSHIBA CORP**

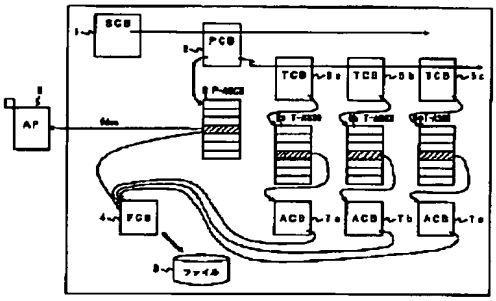
(72) Inventor: **IDE SHUNICHI**

(54) COMPUTER SYSTEM AND FILE ACCESS CONTROL METHOD

(57) Abstract:

PROBLEM TO BE SOLVED: To provide the computer system which can handle a file descriptor as a resource in process units and use the file descriptor not exclusively.

SOLUTION: When a process 9 makes a request to open a file 8, a file descriptor is assigned to the file 8 and sent back to the process 9 at the request source; when some thread of the process 9 makes a request to access the file 8 for the 1st time by using the file descriptor, access control information regarding the file 8 including a pointer is held by access control blocks 7a-7c. The access control blocks are secured on a memory while made to correspond to the thread, and then the file descriptor is handled as a resource in process units; and threads have the access control blocks 7a-7d individually and then threads in the same process use the file descriptor not exclusively.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平9－69060

(43)公開日　平成９年(1997)３月11日

(51)Int.Cl.[6]	識別記号	庁内整理番号	Ｆ Ｉ		技術表示箇所
Ｇ０６Ｆ　12/00	５３５		Ｇ０６Ｆ　12/00	５３５Ｚ	

審査請求　未請求　請求項の数２　ＯＬ　（全７頁）

(21)出願番号　特願平7－223824

(22)出願日　平成７年(1995)８月31日

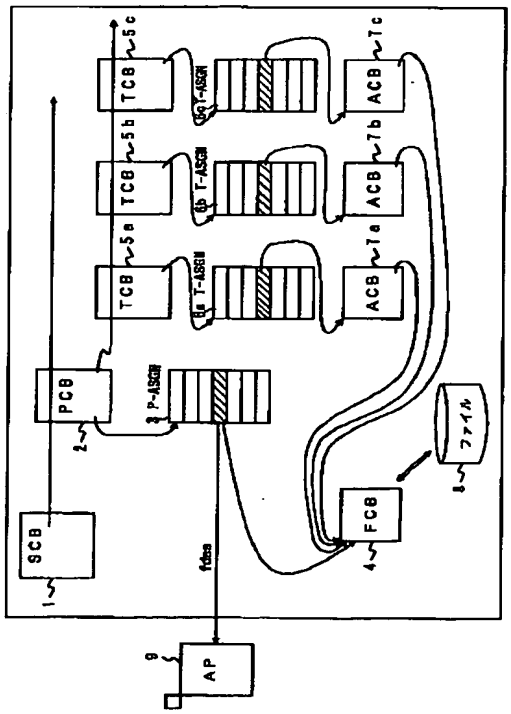
(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地

(72)発明者　井手　俊一
東京都青梅市末広町２丁目９番地　株式会社東芝青梅工場内

(74)代理人　弁理士　鈴江　武彦

(54)【発明の名称】　計算機システムおよびファイルアクセス制御方法

(57)【要約】
【課題】ファイル記述子をプロセス単位の資源として扱うとともに、このファイル記述子を排他せずに使用することのできる計算機システムを提供する。
【解決手段】プロセス９がファイル８のオープンを要求したときに、ファイル９にファイル記述子を割り当てて、そのファイル記述子を要求元のプロセス９に返送し、プロセス９のいずれかのスレッドがそのファイル記述子を用いてファイル８へのアクセスを初めて要求したときに、ポインタを含むファイル８に関するアクセス制御情報を保持するアクセス制御ブロック７ａ～７ｃをそのスレッドに対応させてメモリに確保することにより、ファイル記述子をプロセス単位の資源として扱うとともに、スレッドそれぞれが個別にアクセス制御ブロック７ａ～７ｃをもつことによって、同一プロセス内のスレッドすべてがファイル記述子を排他することなく使用することを特徴とする。



1

【特許請求の範囲】

【請求項１】　一つのプロセスがＣＰＵへの割り付け単位となるスレッドを複数もつことのできるマルチスレッド環境を提供する計算機システムであって、前記スレッドそれぞれが、ファイルそれぞれに固有に割り当てられたファイル記述子を用いてファイルへのアクセスを行なう計算機システムにおいて、

前記プロセスがファイルのオープンを要求したときに、そのオープンの要求されたファイルにファイル記述子を割り当てて、そのファイル記述子を前記要求元のプロセスに返送する手段と、

前記プロセスのいずれかのスレッドが前記割り当てられたファイル記述子を用いてそのファイルへのアクセスを初めて要求したときに、アクセスポインタを含むそのファイルに関するアクセス制御情報を保持するアクセス制御ブロックをそのスレッドに対応させて前記メモリに確保する手段とを具備し、

前記ファイル記述子をプロセス単位の資源として扱うとともに、前記スレッドそれぞれが個別にアクセス制御ブロックをもつことによって、同一プロセス内のスレッドすべてが前記ファイル記述子を排他することなく使用することを特徴とする計算機システム。

【請求項２】　一つのプロセスがＣＰＵへの割り付け単位となるスレッドを複数もつことのできるマルチスレッド環境を提供する計算機システムであって、これらのスレッドそれぞれがファイルそれぞれに固有に割り当てられたファイル記述子を用いてファイルへのアクセスを行なう計算機システムのファイルアクセス制御方法において、

前記プロセスがファイルのオープンを要求したときに、そのオープンの要求されたファイルにファイル記述子を割り当てて、そのファイル記述子を前記要求元のプロセスに返送するステップと、

前記プロセスのいずれかのスレッドが前記割り当てられたファイル記述子を用いてそのファイルへのアクセスを初めて要求したときに、アクセスポインタを含むそのファイルに関するアクセス制御情報を保持するアクセス制御ブロックをそのスレッドに対応させて前記メモリに確保するステップとを具備し、

前記ファイル記述子をプロセス単位の資源として扱うとともに、前記スレッドそれぞれが個別にアクセス制御ブロックをもつことによって、同一プロセス内のスレッドすべてが前記ファイル記述子を排他することなく使用することを特徴とするファイルアクセス制御方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、マルチスレッド環境を提供する計算機システムおよび同計算機システムに適用して好適なファイルアクセス制御方法に係り、特にファイル記述子をプロセス単位の資源として扱うとともに、同一プロセス内のスレッドすべてがファイル記述子を排他することなく使用することを可能とする計算機システムおよびファイルアクセス制御方法に関する。

2

【０００２】

【従来の技術】近年の計算機システムの普及は目覚ましいものがあり、また、この普及に伴って処理の高速化がますます要求されてきている。そして、この高速化を実現するものの一つとして、複数のプロセッサを有する並列計算機システムが存在し、この並列計算機システム上で主に提供される環境としてマルチスレッド環境が存在する。

【０００３】このマルチスレッド環境で実行されるプロセス、すなわちアプリケーションは、プロセッサへの割り付け単位となるスレッドを複数もつことが可能であり、したがって、同一プロセス内のスレッドが別々のプロセッサで並列に実行することができるため、処理の高速化が図られることになる。

【０００４】ここで、図６を参照して従来のマルチスレッド環境下でのファイルアクセスの制御原理について説明する。プロセス（ＡＰ）１７が生成されたとき、システムは、プロセス制御ブロック（ＰＣＢ）およびアサインテーブル（ＡＳＧＮ）をメモリに確保して、このプロセス制御ブロックをシステム制御ブロック（ＳＣＢ）１１に登録する。ここでは、プロセス制御ブロック１２ａおよびアサインテーブル１３がプロセス１７に対応して確保されたものとする。

【０００５】このシステム制御ブロック１１は、システム上で稼働するプロセス全体を制御するために用いられるものである。また、プロセス制御ブロック１２ａ～１２ｃおよびアサインテーブル１３は、生成されたプロセスそれぞれに確保されるものである。そして、プロセス制御ブロックは、そのプロセスの状態管理などに用いられるものであり、アサインテーブルは、後述するアクセス制御ブロック（ＡＣＢ）を管理するために用いられるものである。

【０００６】ここで、このプロセス１７がファイル１６のオープンを要求すると、システムは、そのファイル１６に対応させてアクセス制御ブロック１４をメモリに確保して、このアクセス制御ブロック１４をアサインテーブル１３のいずれかのエントリに設定する。そして、この設定したエントリをファイル記述子（ｆｄｅｓ）としてプロセス１７に返送する。

【０００７】このアクセス制御ブロック１４には、有編成ファイルにおけるアクセスポインタやレコードロック情報、無編成ファイルにおけるカレントアクセス位置などといったアクセス制御情報が保持される。なお、このようなシステムでは、ファイルそれぞれにファイル制御ブロック（ＦＣＢ）を必要に応じてメモリに確保し、このファイル制御ブロックにファイルの制御情報を保持している。そして、アクセス制御ブロックは、このファイ

ル制御ブロックを参照するためのアドレスなどを保持している。

【０００８】そして、プロセス１７のスレッドそれぞれは、前述のオープン時にシステムから受けとったファイル記述子を用いてファイル１６へのアクセス要求を発行する。

【０００９】これによりシステムは、アクセス制御ブロック１４に基づいたファイルアクセスを実施することができる。このように、マルチスレッド環境で実行されるプロセスでは、一つのファイルに対し、各スレッドが同じファイル記述子を用いてアクセスを行なうため、ファイル記述子をプロセス単位の資源（プロセスコンテクスト）として扱うことができ、ファイルのオープンおよびクローズを、プロセス単位の処理としてそれぞれ初期処理および終了処理で一括して行なうなどの構造化プログラミングを可能としていた。

【００１０】しかしながら、各プロセスが、アクセス制御ブロックを一つのファイルに対して一つしか確保していないために、スレッド相互間でファイル記述子を排他して用いなければならず、したがって、ファイル記述子の排他を考慮したプログラミングを行なわなければならなかった。

【００１１】一方、ファイルのオープンおよびクローズを、プロセス単位の処理として行なうのではなく、各スレッドが、アクセスするファイルについて独自にオープンおよびクローズを行ない、アクセス制御ブロックをスレッドそれぞれに確保するといった方法も存在する。

【００１２】しかしながら、ファイルのオープンおよびクローズを、スレッド単位の処理として行なうと、同一ファイルについてファイル記述子が複数設定されてしまうこととなり、ファイル記述子をプロセス単位の資源として扱うことができなくなるために、プログラム構造をいたずらに複雑にしてしまうとともに、その生産性を著しく低下させてしまう。

【００１３】

【発明が解決しようとする課題】前述したように、従来のマルチスレッド環境下におけるファイルアクセス制御では、複数のスレッドが単一のアクセス制御ブロックを使用するために、プロセス内のスレッドそれぞれが、ファイル記述子を排他的に使用しなければならなかった。

【００１４】また、複数のアクセス制御ブロックを確保する場合には、スレッドごとに同一ファイルをオープンし、スレッドごとに異なるファイル記述子を使用しなければならないため、ファイル記述子がプロセスコンテキスト、すなわちプロセス単位の資源として扱うことはできなかった。

【００１５】本発明は、このような実情に鑑みてなされたものであり、ファイル記述子をプロセス単位の資源として扱うとともに、同一プロセス内のスレッドすべてがファイル記述子を排他することなく使用することを可能

とする計算機システムおよびファイルアクセス制御方法を提供することを目的とする。

【００１６】

【課題を解決するための手段】本発明の計算機システムは、一つのプロセスがＣＰＵへの割り付け単位となるスレッドを複数もつことのできるマルチスレッド環境を提供する計算機システムであって、前記スレッドそれぞれが、ファイルそれぞれに固有に割り当てられたファイル記述子を用いてファイルへのアクセスを行なう計算機システムにおいて、前記プロセスがファイルのオープンを要求したときに、そのオープンの要求されたファイルにファイル記述子を割り当てて、そのファイル記述子を前記要求元のプロセスに返送する手段と、前記プロセスのいずれかのスレッドが前記割り当てられたファイル記述子を用いてそのファイルへのアクセスを初めて要求したときに、アクセスポインタを含むそのファイルに関するアクセス制御情報を保持するアクセス制御ブロックをそのスレッドに対応させて前記メモリに確保する手段とを具備し、前記ファイル記述子をプロセス単位の資源として扱うとともに、前記スレッドそれぞれが個別にアクセス制御ブロックをもつことによって、同一プロセス内のスレッドすべてが前記ファイル記述子を排他することなく使用することを特徴とする。

【００１７】また、本発明のファイルアクセス制御方法は、一つのプロセスがＣＰＵへの割り付け単位となるスレッドを複数もつことのできるマルチスレッド環境を提供する計算機システムであって、これらのスレッドそれぞれがファイルそれぞれに固有に割り当てられたファイル記述子を用いてファイルへのアクセスを行なう計算機システムのファイルアクセス制御方法において、前記プロセスがファイルのオープンを要求したときに、そのオープンの要求されたファイルにファイル記述子を割り当てて、そのファイル記述子を前記要求元のプロセスに返送するステップと、前記プロセスのいずれかのスレッドが前記割り当てられたファイル記述子を用いてそのファイルへのアクセスを初めて要求したときに、アクセスポインタを含むそのファイルに関するアクセス制御情報を保持するアクセス制御ブロックをそのスレッドに対応させて前記メモリに確保するステップとを具備し、前記ファイル記述子をプロセス単位の資源として扱うとともに、前記スレッドそれぞれが個別にアクセス制御ブロックをもつことによって、同一プロセス内のスレッドすべてが前記ファイル記述子を排他することなく使用することを特徴とする。

【００１８】本発明によれば、プロセスがファイルのオープンを要求してきたときに、システムは、そのファイルにファイル記述子を割り当てるとともに、そのファイル記述子を要求元のプロセスに返送する。したがって、この時点では、アクセス制御ブロックの確保は行なわれない。

【００１９】次に、プロセス内のスレッドがそのファイル記述子を用いて初めてアクセスを行なった際、システムは、このスレッドに対応させて、そのファイルについてのアクセス制御ブロックを確保する。すなわち、各スレッドは、それぞれ固有のアクセス制御ブロックをもつことになる。

【００２０】これにより、ファイル記述子をプロセス単位の資源として取り扱うことができ、かつ各スレッド相互間でこのファイル記述子を排他して使用するといったことが不要となる。

【００２１】

【発明の実施の形態】以下、図面を参照して本発明の一実施形態を説明する。図１は同実施形態におけるファイルアクセスの制御原理について説明するための概念図である。

【００２２】同実施形態において、プロセス（ＡＰ）９が生成されると、システムは、プロセス制御ブロック（ＰＣＢ）２およびプロセスアサインテーブル（Ｐ－ＡＳＧＮ）３をメモリに確保して、このプロセス制御ブロック２をシステム制御ブロック（ＳＣＢ）１に登録する。

【００２３】このシステム制御ブロック１は、システム上で稼働するプロセス全体を制御するために用いられるものである。また、プロセス制御ブロック２は、プロセス９の状態管理などに用いられものであり、プロセスアサインテーブル３は、ファイル制御ブロック（ＦＣＢ）４を参照するときに用いられるものである。なお、ファイル制御ブロックとは、ファイルそれぞれに対応して必要に応じてメモリに確保されるものであり、ファイルの属性などを含む制御情報を保持するものである。

【００２４】また、プロセス９によりスレッドが生成されると、システムは、スレッド制御ブロック５ａ～５ｃおよびスレッドアサインテーブル６ａ～６ｃをメモリに確保して、このスレッド制御ブロック５ａ～５ｃをプロセス制御ブロック２に登録する。なお、このスレッド制御ブロック５ａ～５ｃおよびスレッドアサインテーブル６ａ～６ｃは、それぞれ一対にして確保され、各スレッド制御ブロック５ａ～５ｃは、それぞれに対応したスレッドアサインテーブル６ａ～６ｃのアドレスを保持している。

【００２５】次に、このプロセス９がファイル８のオープンを要求すると、システムは、そのファイル８に対応したファイル制御ブロック４のアドレスをプロセスアサインテーブル３のいずれかのエントリに設定し、この設定したエントリをファイル記述子としてプロセス９に返送する。

【００２６】そして、プロセス９のスレッドが、このファイル記述子を用いてファイル８へのアクセスを要求すると、システムは、このファイル記述子が、そのスレッドに対応して設けられたスレッドアサインテーブル（ここでは６ａとする）内で有効であるかどうかを判定する。

【００２７】このスレッドアサインテーブル６ａの検索は、以下の手順で行なわれる。すなわち、スレッドからのアサイン要求が発生すると、システムは、プロセスＩＤとスレッドＩＤとを取得して、まずプロセスＩＤからプロセス制御ブロック２を知得する。次に、このプロセス制御ブロック２とスレッドＩＤとからスレッド制御ブロック５ａを知得する。そして、このスレッド制御ブロック５ａの保持するスレッドアサインテーブル６ａのアドレスによりスレッドアサインテーブル６ａを検索する。

【００２８】そして、システムは、この検索したスレッドアサインテーブル６ａにアクセスの要求されたファイル８に対応したアクセス制御ブロック７ａが存在するかどうかを検査することにより、そのファイル記述子が、そのスレッドアサインテーブル６ａ内で有効であるかどうかを判定する。

【００２９】なお、このアクセス制御ブロックとは、有編成ファイルにおけるアクセスポインタやレコードロック情報、無編成ファイルにおけるカレントアクセス位置などといったアクセス制御情報を保持するものであり、各ファイルに対応して確保されるものである。また、アクセス制御ブロックは、ファイル制御ブロックを参照するためのアドレスなども保持している。

【００３０】ここで、このファイル記述子が有効であった場合には、そのスレッドアサインテーブル６ａに設定されたアクセス制御ブロック７ａに基づいて、アクセス処理を実施する。一方、ファイル記述子が有効でなかった場合には、さらにそのファイル記述子がプロセスアサインテーブル３内で有効であるかどうかを判定する。

【００３１】ここで、このファイル記述子がプロセスアサインテーブル３内で有効でない場合には、ファイルのオープンが行なわれていないこととなるため、エラーとしてプロセス９にその旨を返答する。

【００３２】一方、このファイル記述子がプロセスアサインテーブル３内で有効であった場合には、アクセス制御ブロック７ａをメモリに確保してスレッドアサインテーブル６ａに設定するとともに、そのアクセス制御ブロック７ａを用いてアクセス処理を実施する。

【００３３】すなわち、各スレッドがアクセス制御ブロックをそれぞれにもつために、ファイル記述子をプロセス単位の資源として扱うことができ、かつファイル記述子の排他をまったく意識する必要がない。

【００３４】次に図２乃至図５を参照して同実施形態の動作手順を説明する。図２は同実施形態のプロセスが生成されたときの動作を説明するフローチャートである。

【００３５】プロセス９が生成されると、システムは、プロセス制御ブロック２をメモリに確保して（図２のステップＡ１）、ついでプロセスアサインテーブル３をメ

モリに確保する（図２のステップＡ２）。

【００３６】図３は同実施形態のプロセスによりスレッドが生成されたときの動作を説明するフローチャートである。プロセス９によりスレッドが生成されると、システムは、スレッド制御ブロック５ａ～５ｃをメモリに確保して（図３のステップＢ１）、ついでこれらスレッド制御ブロック５ａ～５ｃそれぞれと一対にしてスレッドアサインテーブル６ａ～６ｃをメモリに確保する（図３のステップＢ２）。

【００３７】図４は同実施形態のプロセスがファイルのオープンを要求しときの動作を説明するフローチャートである。プロセス９が、ファイル８のオープンを要求すると、システムは、プロセスＩＤからプロセス制御ブロック２を知得し、さらにこのプロセス制御ブロック２からプロセスアサインテーブル３を知得する（図４のステップＣ１）。そして、システムは、このオープン要求のあったファイル８に対応するファイル制御ブロック４がメモリに確保されているかどうかを判定し（図４のステップＣ２）、確保されていないときに、ファイル制御ブロック４をメモリ上に確保する（図４のステップＣ３）。

【００３８】次に、システムは、プロセスアサインテーブル３のいずれかのエントリを確保して（図４のステップＣ４）、ファイル制御ブロック４をこのエントリに設定するともに、プロセス９により指定されたオープンモードを格納する（図４のステップＣ５）。そして、この確保したエントリのエントリ番号をファイル記述子として、プロセス９に返送する（図４のステップＣ６）。

【００３９】図５は同実施形態のスレッドがファイル記述子を用いてファイルのアクセスを要求したときの動作を説明するフローチャートである。プロセス９のスレッドが、ファイル記述子を用いてファイル８へのアクセスを要求すると、システムは、まずプロセスＩＤからプロセス制御ブロック２を知得し、さらにこのプロセス制御ブロック２からプロセスアサインテーブル３を知得する（図５のステップＤ１）。次にシステムは、スレッドＩＤからスレッド制御ブロック５ａを知得し、さらにこのスレッド制御ブロック５ａからスレッドアサインテーブル６ａを知得する（図５のステップＤ２）。

【００４０】ここでシステムは、ファイル記述子が、そのスレッドアサインテーブル６ａ内で有効であるかどうかを判定し（図５のステップＤ３）、有効でない場合には、さらにそのファイル記述子がプロセスアサインテーブル３内で有効であるかどうかを判定する（図５のステップＤ４）。

【００４１】ここで、このファイル記述子がプロセスアサインテーブル３内で有効でない場合には、ファイルのオープンが行なわれていないこととなるため、エラーとしてプロセス９にその旨を返答する。

【００４２】一方、このファイル記述子がプロセスアサインテーブル３内で有効であった場合には、アクセス制御ブロック７ａをメモリに確保してスレッドアサインテーブル６ａに設定するとともに（図５のステップＤ５～ステップＤ６）、そのアクセス制御ブロック７ａを用いてアクセス処理を実施する（図５のステップＤ７）。これにより、ファイル記述子をプロセス単位の資源として扱うことができ、かつファイル記述子の排他をまったく意識する必要がない。

【００４３】

【発明の効果】以上詳述したように、本発明によれば、ファイル記述子をプロセス単位の資源として扱えるとともに、スレッドそれぞれが個別にアクセス制御ブロックをもつことによって、同一プロセス内のスレッドすべてがファイル記述子を排他することなく使用することができることとなる。

【図面の簡単な説明】

【図１】本発明の実施形態におけるファイルアクセスの制御原理について説明するための概念図。

【図２】同実施形態のプロセスが生成されたときの動作を説明するフローチャート。

【図３】同実施形態のプロセスによりスレッドが生成されたときの動作を説明するフローチャート。

【図４】同実施形態のプロセスがファイルのオープンを要求しときの動作を説明するフローチャート。

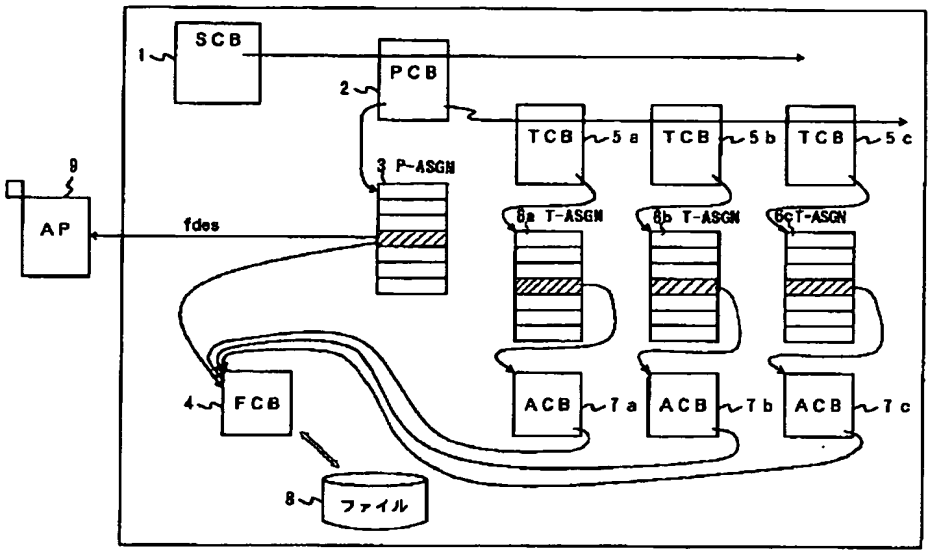
【図５】同実施形態のスレッドがファイル記述子を用いてファイルのアクセスを要求しときの動作を説明するフローチャート。

【図６】従来のマルチスレッド環境下でのファイルアクセスの制御原理を説明するための概念図。

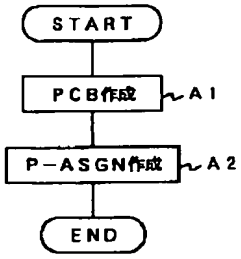
【符号の説明】

１…システム制御ブロック、２…プロセス制御ブロック、３…プロセスアサインテーブル、４…ファイル制御ブロック、５ａ，５ｂ，５ｃ…スレッド制御ブロック、６ａ，６ｂ，６ｃ…スレッドアサインテーブル、７ａ，７ｂ，７ｃ…アクセス制御ブロック、８…ファイル、９…プロセス。

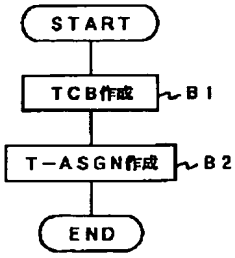
【図１】


SCB
1
PCB
2
TCB
5a
TCB
5b
TCB
5c
3 P-ASGN
9
AP
fdes
8a T-ASGN
8b T-ASGN
8cT-ASGN
4
FCB
ACB
7a
ACB
7b
ACB
7c
8
ファイル

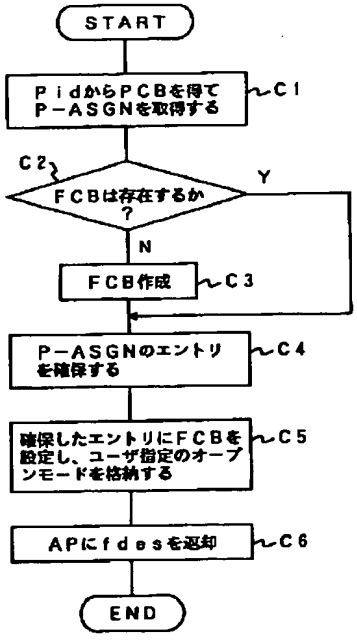
【図２】


START
PCB作成
A1
P－ASGN作成
A2
END

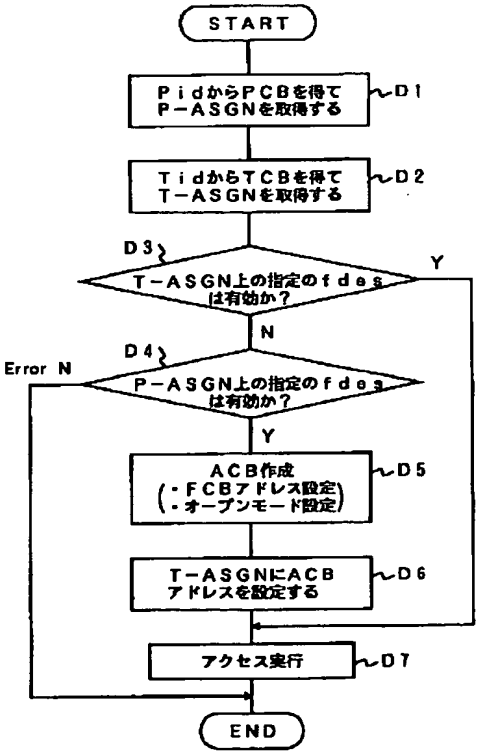
【図３】


START
TCB作成
B1
T－ASGN作成
B2
END

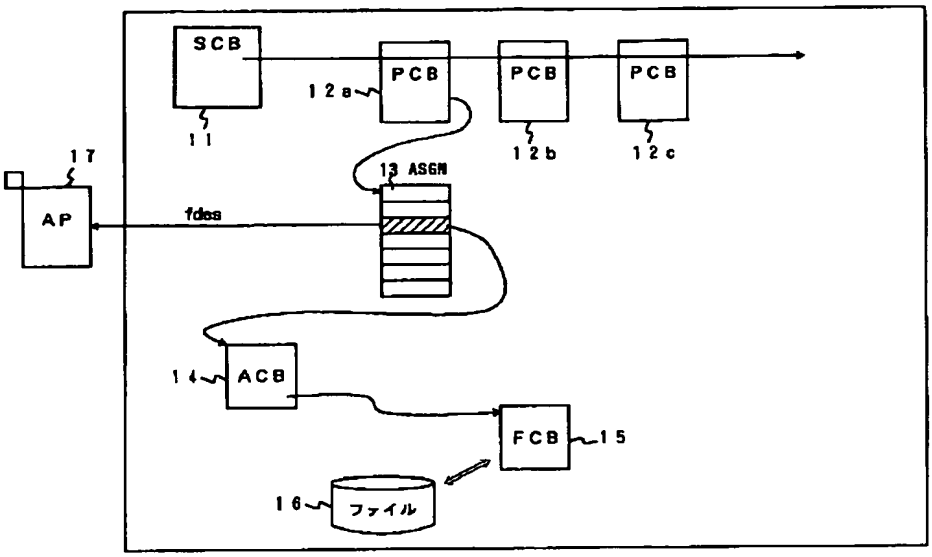
【図４】


START
PidからPCBを得て
P－ASGNを取得する
C1
C2
FCBは存在するか
？
Y
N
FCB作成
C3
P－ASGNのエントリ
を確保する
C4
確保したエントリにFCBを
設定し、ユーザ指定のオープ
ンモードを格納する
C5
APにfdesを返却
C6
END

【図５】


START
PidからPCBを得て
P－ASGNを取得する
D1
TidからTCBを得て
T－ASGNを取得する
D2
D3
T－ASGN上の指定のfdes
は有効か？
Y
N
Error N
D4
P－ASGN上の指定のfdes
は有効か？
Y
ACB作成
(・FCBアドレス設定)
(・オープンモード設定)
D5
T－ASGNにACB
アドレスを設定する
D6
アクセス実行
D7
END

【図６】


SCB
11
12a
PCB
12b
PCB
12c
PCB
13 ASGN
17
AP
fdes
14
ACB
FCB
15
16
ファイル